



ITW
Duc
S

Patent
Attorney's Docket No. 0320-002

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

In re Patent Application of)
)
Frank A. Hunleth et al.) Group Art Unit:
)
Application No.: 10/768,234) Examiner:
)
Filed: January 30, 2004)
)
For: METHODS AND SYSTEMS FOR)
GENERATING A ZOOMABLE)
GRAPHICAL USER INTERFACE)

PETITION UNDER 37 C.F.R. 1.182

Mail Stop Petitions
Commissioner for Patents
Alexandria, VA 22313-1450

Sir:

In response to the Notice of Incomplete Nonprovisional Application dated June 15, 2004 (copy attached hereto), Applicants respectfully submit the following petition.

1. The Notice of Incomplete Nonprovisional Application indicates that a filing date has not been accorded the above-identified application due to a lack of receipt by the PTO of a claim and/or abstract associated with the application filed on January 30, 2004.

2. The above-identified application, on page 2, paragraph 0001, incorporates by reference the disclosure of U.S. Provisional Patent Application No. 60/495,998.

3. U.S. Provisional Patent Application No. 60/495,998 includes both claims and an abstract. A certified copy of this application is attached to this petition. Accordingly, it is

respectfully submitted that U.S. Patent Application No. 10/768,234 also contained claims and an abstract by virtue of this express incorporation by reference.

4. A telephone discussion with Ms. Gena Jones of the Petitions Office indicated that the above-recited facts could form the basis for a grantable petition under 37 C.F.R. 1.182 requesting that the original filing date be granted.

5. The requisite petition fee pursuant to 37 C.F.R. 1.17(h) of \$130.00 is submitted herewith.

6. Accordingly, Applicants respectfully submit that the above-identified application is entitled to the filing date of January 30, 2004, hereby request the U.S. Patent and Trademark Office to so indicate by GRANTING this petition.

Should there be any questions regarding this petition, kindly contact the undersigned at (540)-361-1863. In the event that this petition should have been styled as a petition under 37 C.F.R. 1.181, it is respectfully requested that this petition be alternately considered thereunder.

Respectfully submitted,

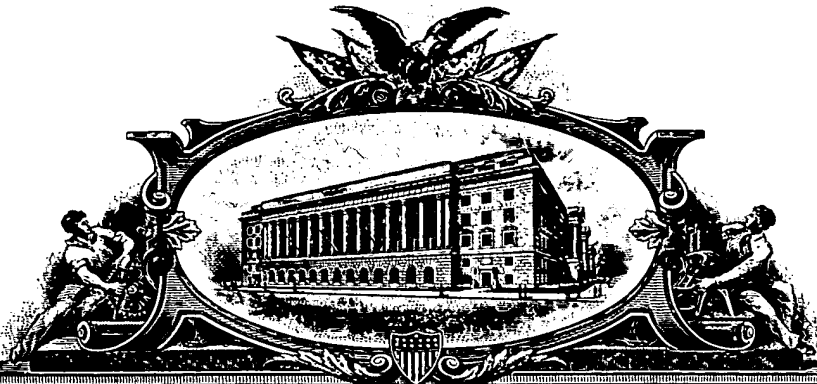
POTOMAC PATENT GROUP PLLC

By: 

Steven M. duBois

Registration No. 35,023

Date: July 27, 2004



THE UNITED STATES OF AMERICA

TO ALL TO WHOM THESE PRESENTS SHALL COME:

UNITED STATES DEPARTMENT OF COMMERCE

United States Patent and Trademark Office

July 15, 2004

**THIS IS TO CERTIFY THAT ANNEXED HERETO IS A TRUE COPY FROM
THE RECORDS OF THE UNITED STATES PATENT AND TRADEMARK
OFFICE OF THOSE PAPERS OF THE BELOW IDENTIFIED PATENT
APPLICATION THAT MET THE REQUIREMENTS TO BE GRANTED A
FILING DATE UNDER 35 USC 111.**

APPLICATION NUMBER: 60/495,998

FILING DATE: August 18, 2003



**By Authority of the
COMMISSIONER OF PATENTS AND TRADEMARKS**

H. L. Jackson

H. L. JACKSON

Certifying Officer

08/18/03

13146 U.S. PTO



PTO/SB/16 (08-03)

Approved for use through 07/31/2006. OMB 0651-0032

U.S. Patent and Trademark Office; U.S. DEPARTMENT OF COMMERCE

Under the Paperwork Reduction Act of 1995, no persons are required to respond to a collection of information unless it displays a valid OMB control number.

PROVISIONAL APPLICATION FOR PATENT COVER SHEET

This is a request for filing a PROVISIONAL APPLICATION FOR PATENT under 37 CFR 1.53(c).

Express Mail Label No.

ER 431542356 45

PTO
60/495998

INVENTOR(S)					
Given Name (first and middle [if any])		Family Name or Surname		Residence (City and either State or Foreign Country)	
Frank A. Negar		Hunleth Moshiri		Rockville, MD Bethesda, MD	
Additional inventors are being named on the <u>1</u> separately numbered sheets attached hereto					
TITLE OF THE INVENTION (500 characters max)					
A Control Framework with a Zoomable Graphical User Interface for Organizing, Selecting and Launching Media Items					
Direct all correspondence to: CORRESPONDENCE ADDRESS					
<input type="checkbox"/> Customer Number: 					
OR					
<input checked="" type="checkbox"/> Firm or Individual Name		Hillcrest Communications, Inc.			
Address		15245 Shady Grove Road			
Address		Suite 450			
City		Rockville		State	MD
Country		USA		Zip	20850
		Telephone	2403860600	Fax	240-386-0601
ENCLOSED APPLICATION PARTS (check all that apply)					
<input checked="" type="checkbox"/> Specification Number of Pages <u>64</u>		<input type="checkbox"/> CD(s), Number _____			
<input checked="" type="checkbox"/> Drawing(s) Number of Sheets <u>17</u>		<input type="checkbox"/> Other (specify) _____			
<input type="checkbox"/> Application Date Sheet. See 37 CFR 1.76					
METHOD OF PAYMENT OF FILING FEES FOR THIS PROVISIONAL APPLICATION FOR PATENT					
<input checked="" type="checkbox"/> Applicant claims small entity status. See 37 CFR 1.27.				FILING FEE Amount (\$)	
<input type="checkbox"/> A check or money order is enclosed to cover the filing fees.				<div style="border: 1px solid black; padding: 10px; text-align: center;">\$80.00</div>	
<input type="checkbox"/> The Director is hereby authorized to charge filing fees or credit any overpayment to Deposit Account Number: _____					
<input checked="" type="checkbox"/> Payment by credit card. Form PTO-2038 is attached.					
The invention was made by an agency of the United States Government or under a contract with an agency of the United States Government.					
<input checked="" type="checkbox"/> No.					
<input type="checkbox"/> Yes, the name of the U.S. Government agency and the Government contract number are: _____					

[Page 1 of 2]

Respectfully submitted

SIGNATURE

TYPED or PRINTED NAME Frank A. Hunleth

TELEPHONE 240 386 0600 x235

Date August 18, 2003

REGISTRATION NO. _____

(if appropriate)

Docket Number: _____

USE ONLY FOR FILING A PROVISIONAL APPLICATION FOR PATENT

This collection of information is required by 37 CFR 1.51. The information is required to obtain or retain a benefit by the public which is to file (and by the USPTO to process) an application. Confidentiality is governed by 35 U.S.C. 122 and 37 CFR 1.14. This collection is estimated to take 8 hours to complete, including gathering, preparing, and submitting the completed application form to the USPTO. Time will vary depending upon the individual case. Any comments on the amount of time you require to complete this form and/or suggestions for reducing this burden, should be sent to the Chief Information Officer, U.S. Patent and Trademark Office, U.S. Department of Commerce, P.O. Box 1450, Alexandria, VA 22313-1450. DO NOT SEND FEES OR COMPLETED FORMS TO THIS ADDRESS. SEND TO: Mail Stop Provisional Application, Commissioner for Patents, P.O. Box 1450, Alexandria, VA 22313-1450.

If you need assistance in completing the form, call 1-800-PTO-9199 and select option 2.

PROVISIONAL APPLICATION COVER SHEET
Additional Pag

PTO/SB/16 (08-03)

Approved for use through 07/31/2006. OMB 0651-0032

U.S. Patent and Trademark Office; U.S. DEPARTMENT OF COMMERCE

Under the Paperwork Reduction Act of 1995, no persons are required to respond to a collection of information unless it displays a valid OMB control number.

Docket Number

INVENTOR(S)/APPLICANT(S)		
Given Name (first and middle [if any])	Family or Surname	Residence (City and either State or Foreign Country)
William J.	Napier	Germantown, MD
Daniel S.	Simpkins	Bethesda, MD
Frank J.	Wroblewski	Gaithersburg, MD

[Page 2 of 2]

Number 2 of 2

WARNING: Information on this form may become public. Credit card information should not be included on this form. Provide credit card information and authorization on PTO-2038.

UNITED STATES PATENT APPLICATION

OF

Frank A. Hunleth

Negar Moshiri

William J. Napier

Daniel S. Simpkins

Frank Wroblewski

FOR

**A CONTROL FRAMEWORK WITH A ZOOMABLE GRAPHICAL USER INTERFACE
FOR ORGANIZING, SELECTING AND LAUNCHING MEDIA ITEMS**

**A CONTROL FRAMEWORK WITH A ZOOMABLE GRAPHICAL USER
INTERFACE FOR ORGANIZING, SELECTING AND LAUNCHING MEDIA ITEMS**

BACKGROUND

[0002] The present invention relates generally to the design and operation of graphical user interfaces with the basic building blocks of point, click, scroll, hover and zoom and, more particularly, to graphical user interfaces associated with media items optionally using a free-space pointing remote.

[0003] Technologies associated with the communication of information have evolved rapidly over the last several decades. Television, cellular telephony, the Internet and optical communication techniques (to name just a few things) combine to inundate consumers with available information and entertainment options. Taking television as an example, the last three decades have seen the introduction of cable television service, satellite television service, pay-per-view movies and video-on-demand. Whereas television viewers of the 1960s could typically receive perhaps four or five over-the-air TV channels on their television sets, today's TV watchers have the opportunity to select from hundreds and potentially thousands of channels of shows and information. Video-on-demand technology, currently used primarily in hotels and the like, provides the potential for in-home entertainment selection from among thousands of movie titles.

[0004] The technological ability to provide so much information and content to end users provides both opportunities and challenges to system designers and service providers. One challenge is that while end users typically prefer having more choices rather than fewer, this preference is counterweighted by their desire that the selection process be both fast and simple. Unfortunately, the development of the interfaces by which end users access media items has resulted in interfaces in which the selection process is neither fast nor simple. Consider again the example of television programs. When television was in its infancy, determining which program to watch was a relatively simple process primarily due to the small number of choices. One would consult a printed guide which was formatted, for example, as series of columns and rows which showed the correspondence between (1) nearby television channels, (2) programs being transmitted on those channels and (3) date and time. The television was tuned to the desired channel by adjusting a tuner knob and the viewer watched the selected program. Later, remote control devices were introduced that permitted viewers to tune the television from a distance. This addition to the user-television interface created the phenomenon known as "channel surfing" whereby a viewer could rapidly view short segments being broadcast on a number of channels to quickly learn what programs were available at any given time.

[0005] Despite the fact that the number of channels and amount of viewable content has dramatically increased, the generally available user interface and control device options and framework for televisions has not changed much over the last 30 years. Printed guides are still the most prevalent mechanism for conveying programming information. The multiple button remote control with simple up and down arrows is still the most prevalent

channel/content selection mechanism. The reaction of those who design and implement the TV user interface to the increase in available media content has been a straightforward extension of the existing interface objects. Thus, the number of rows and columns in the printed guides has been increased to accommodate more channels. The number of buttons on the remote control devices has been increased to support additional functionality and content handling, e.g., as shown in Figure 1. However, this approach has significantly increased both the time required for a viewer to review the available information and the complexity of actions required to implement a selection. Arguably, the cumbersome nature of the existing interface has hampered commercial implementation of some services, e.g., video-on-demand, since consumers are resistant to new services that will add complexity to an interface that they view as already too slow and complex. [0006]

In addition to increases in bandwidth and content, the user interface bottleneck problem is being exacerbated by the aggregation of technologies. Consumers are reacting positively to having the option of buying integrated systems rather than a number of segregable components. A good example of this trend is the combination television/VCR/DVD in which three previously independent components are frequently sold today as an integrated unit. This trend is likely to continue, potentially with an end result that most if not all of the communication devices currently found in the household being packaged as an integrated unit, e.g., a television/VCR/DVD/internet access/radio/stereo unit. With more aggregation comes the potential for more complexity in the user interface. For example, when so-called "universal" remote units were introduced, e.g., to combine the functionality of TV remote units and VCR remote units, the number of buttons on these universal remote units was typically more than

the number of buttons on either the TV remote unit or VCR remote unit individually. This added number of buttons makes it very difficult to control anything but the simplest aspects of a TV or VCR without hunting for the right button on the remote. Many times, these universal remotes do not provide enough buttons to access many expert level controls or features unique to certain TVs. In these cases, the original device remote unit is still needed, and the original hassle of handling multiple remotes remains due to user interface issues arising from the complexity of aggregation. Some remote units have addressed this problem by adding "soft" buttons that can be programmed with the expert commands. These soft buttons sometimes have accompanying LCD displays to indicate their action. These too have the flaw that they are difficult to use without looking away from the TV to the remote control. Yet another flaw in these remote units is the use of modes in an attempt to reduce the number of buttons. In these "moded" universal remote units, a special button exists to select whether the remote should communicate with the TV, DVD player, cable set-top box, VCR, etc. This causes many usability issues including sending commands to the wrong device, forcing the user to look at the remote to make sure that it is in the right mode, and it does not provide any simplification to the integration of multiple devices. The most advanced of these universal remote units provide some integration by allowing the user to program sequences of commands to multiple devices into the remote. This is such a difficult task that many users hire professional installers to program their universal remote units.

[0007] Some attempts have also been made to modernize the screen interface between end users and media systems. Electronic program guides (EPGs) have been developed and

implemented to replace the afore-described media guides. Early EPGs provided what was essentially an electronic replica of the printed media guides. For example, cable service operators have provided analog EPGs wherein a dedicated channel displays a slowly scrolling grid of the channels and their associated programs over a certain time horizon, e.g., the next two hours. Scrolling through even one hundred channels in this way can be tedious and is not feasibly scalable to include significant additional content deployment, e.g., video-on-demand. More sophisticated digital EPGs have also been developed. In digital EPGs, program schedule information, and optionally applications/system software, is transmitted to dedicated EPG equipment, e.g., a digital set-top box (STB). Digital EPGs provide more flexibility in designing the user interface for media systems due to their ability to interpose one or more interface layers between the user and the selection of the media items to be viewed. An example of such an interface can be found in U.S. Patent No. 6,421,067 to Kamen et al., the disclosure of which is incorporated here by reference. Figure 2 depicts a GUI described in the '067 patent. Therein, according to the Kamen et al. patent, a first column 190 lists program channels, a second column 191 depicts programs currently playing, a column 192 depicts programs playing in the next half-hour, and a fourth column 193 depicts programs playing in the half hour after that. The baseball bat icon 121 spans columns 191 and 192, thereby indicating that the baseball game is expected to continue into the time slot corresponding to column 192. However, text block 111 does not extend through into column 192. This indicates that the football game is not expected to extend into the time slot corresponding to column 192. As can be seen, a pictogram 194 indicates that after the football game, ABC will be showing a horse race. The icons shown in Figure 2 can be actuated using a cursor, not

shown, to implement various features, e.g., to download information associated with the selected programming. Other digital EPGs and related interfaces are described, for example, in U.S. Patent Nos. 6,314,575, 6,412,110, and 6,577,350, the disclosures of which are also incorporated here by reference.

[0008] However, the interfaces described above suffer from, among other drawbacks, an inability to easily scale between large collections of media items and small collections of media items. For example, interfaces which rely on lists of items may work well for small collections of media items, but are tedious to browse for large collections of media items. Interfaces which rely on hierarchical navigation (e.g., tree structures) may be more speedy to traverse than list interfaces for large collections of media items, but are not readily adaptable to small collections of media items. Additionally, users tend to lose interest in selection processes wherein the user has to move through three or more layers in a tree structure. For all of these cases, current remote units make this selection processor even more tedious by forcing the user to repeatedly depress the up and down buttons to navigate the list or hierarchies. When selection skipping controls are available such as page up and page down, the user usually has to look at the remote to find these special buttons or be trained to know that they even exist.

[0009] Accordingly, it would be desirable to provide techniques and systems which simplify the control and screen interface between users and media systems as well as accelerate the selection process, while at the same time permitting service providers to take advantage of the increases in available bandwidth to end user equipment by facilitating the supply of a large number of media items and new services to the user. Moreover, it would be

desirable to provide interfaces which supply an easy and fast selection experience regardless of the size(s) of the media item collection(s) being browsed.

SUMMARY

[0010] Systems and methods according to the present invention address these needs and others by providing a total user interface framework covering both the control and graphical display of media items which then provides for easy and rapid selection of media items. Control of the framework employs a free space pointing device that includes a minimal set of buttons and scroll wheel for pointing, clicking and scrolling through selections on an associated graphical user interface. This graphical user interface provides feedback to the user through the use of an on-screen mouse pointer, graphical animations when the pointer hovers over selections, and zooming into and out of selections to smoothly navigate between overview and detail screens. Exemplary embodiments of the present invention employ images, zooming for increased/decreased levels of detail and continuity of GUI objects which permit easy navigation by a user. Graphical user interfaces according to the present invention organize media item selections on a virtual surface. Similar selections can be grouped together. Initially, the interface presents a zoomed out view of the surface, and in most cases, the actual selections will not be visible in full detail at this level. As the user zooms progressively inward, more details are revealed concerning the media item groups or selections. At different zoom levels, different controls are available so that the user can play groups of selections, individual selections, or go to another part of the virtual surface to browse other related media items.

[0011] According to one exemplary embodiment of the present invention, a method for displaying information on a graphical user interface includes the steps of: displaying a first plurality of images at a first magnification level, receiving a first selection indication that identifies a subset of the plurality of images, displaying a first zoomed version of the selected subset of the plurality of images at a second magnification level, receiving a second selection indication that identifies an image within the subset of images, and displaying a second zoomed version of the identified image, wherein the second zoomed version of the identified image includes at least one user-actuable control element. The control in this example would come from a free-space pointing device using a gyroscope or other mechanism to define both screen position and motion vector to determine the particular command desired. A set of buttons is also included for the “click” or select case as well as a “back” button.

[0012] According to another exemplary embodiment of the present invention, a graphical user interface includes a first GUI screen which displays a first plurality of images at a first magnification level, means for receiving a first selection indication that identifies a subset of the plurality of images, a second GUI screen which displays a first zoomed version of the selected subset of the plurality of images at a second magnification level, means for receiving a second selection indication that identifies an image within the subset of images, and a third GUI screen which displays a second zoomed version of the identified image, wherein the second zoomed version of the identified image includes at least one user-actuable control element. Zooming graphical user interfaces according to exemplary embodiments of the present invention can contain categories of images nested to an arbitrary depth as well as categories of categories. In the present invention, the user points the remote unit at the

category or categories of interest and depresses the selection button to zoom in or the "back" button to zoom back to the overview.

The user interface incorporates several visual techniques to achieve scaling to the very large. It is the combination of building blocks and techniques that achieves both scalability and ease-of-use. Two techniques that are part of the user interface are "visual memory" for rapid re-visiting and "filters" as first-class visual objects.

Filters provide a way for the user to select from extremely large sets of items. The user remains shielded from the details. The user interface itself remains simple by designing filters as first-class visual objects. This means that the user requires no additional building blocks (point, click, scroll, zoom) to construct and use filters. Filters are constructed by selecting from items in a visual zoomable layout using point and click behavior. The selected filter takes its place among the existing visual objects. In addition to the filter function the filter has all the visual attributes as the other objects. You can point to it, hover over it, and zoom into it by clicking on the filter object. Filter objects can be inserted anywhere a visual object can reside. As a result filters can be intermixed, chained, and nested in a myriad of combinations within the visual environment.

The user interface is a visual experience. In this environment we make use of the inherent capability of the user to remember the location of objects within the visual environment. This is achieved by providing a stable dependable location for zoomable items. This remains true

even if the items were organized because of a filter operation. Each object has a location in the zoomable layout. Once the user has found an object of interest it is natural to remember which direction was taken to locate the object. If that object is of particular interest it is likely that the user will re-visit the item more than once. In doing so the memory of the path to the object will be strengthened. The visual interface look-and-feel provides clues that help the user remember the location of items of interest.

These techniques along with the building blocks work in combination to provide a rich yet simple-to-use interface to a very large set of items.

BRIEF DESCRIPTION OF THE DRAWINGS

[0013] The accompanying drawings illustrate exemplary embodiments of the present invention, wherein:

[0014] FIG. 1 depicts a conventional remote control unit for an entertainment system;

[0015] FIG. 2 depicts a conventional graphical user interface for an entertainment system;

[0016] FIG. 3 depicts an exemplary media system in which exemplary embodiments of the present invention (both display and remote control) can be implemented;

[0017] FIG. 4 shows a system controller of FIG. 3 in more detail;

[0018] FIGS. 5-8 depict a graphical user interface for a media system according to an exemplary embodiment of the present invention;

[0019] FIG. 9 illustrates an exemplary data structure according to an exemplary embodiment of the present invention;

[0020] FIGS. 10(a) and 10(b) illustrate a zoomed out and an zoomed in version of a portion of an exemplary GUI created using the data structure of FIG. 9 according to an exemplary embodiment of the present invention;

[0021] FIG. 11 depicts a doubly linked, ordered list used to generated GUI displays according to an exemplary embodiment of the present invention;

[0022] FIG. 12 depicts a data structure according to another exemplary embodiment of the present invention including a virtual camera for use in resolution independent zooming;

[0023] FIGS. 13(a) and 13(b) show a zoomed out and zoomed in version of a portion of an exemplary GUI which depict semantic zooming according to an exemplary embodiment of the present invention;

[0024] FIGS. 14-18 depict a zoomable graphical user interface according to another exemplary embodiment of the present invention;

[0025] FIG. 19 illustrates an exemplary framework for implementing zoomable graphical user interfaces according to the present invention; and

[0026] FIG. 20 {TBD} shows an exemplary infrastructure in which the present invention can be implemented.

DETAILED DESCRIPTION

[0027] The following detailed description of the invention refers to the accompanying drawings. The same reference numbers in different drawings identify the same or similar

elements. Also, the following detailed description does not limit the invention. Instead, the scope of the invention is defined by the appended claims.

[0028] In order to provide some context for this discussion, an exemplary aggregated media system 200 in which the present invention can be implemented will first be described with respect to Figure 3. Those skilled in the art will appreciate, however, that the present invention is not restricted to implementation in this type of media system and that more or fewer components can be included therein. Therein, an input/output (I/O) bus 210 connects the system components in the media system 200 together. The I/O bus 210 represents any of a number of different mechanisms and techniques for routing signals between the media system components. For example, the I/O bus 210 may include an appropriate number of independent audio "patch" cables that route audio signals, coaxial cables that route video signals, two-wire serial lines or infrared or radio frequency transceivers that route control signals, or other routing mechanisms that route other types of signals.

[0029] In this exemplary embodiment, the media system 200 includes a television/monitor 212, a video cassette recorder (VCR) 214, digital video disk (DVD) recorder/playback device 216, audio/video tuner 218 and compact disk player 220 coupled to the I/O bus 210. The VCR 214, DVD 216 and compact disk player 220 may be single disk or single cassette devices, or alternatively may be multiple disk or multiple cassette devices. They may be independent units or integrated together. In addition, the media system 200 includes a microphone/speaker system 222, video camera 224 and a wireless I/O control device 226. In the invention, the wireless I/O control device 226 is a media system remote control unit that supports free space pointing, has a minimal number of buttons to support

navigation, and communicates with the entertainment system 200 through RF signals. In another exemplary embodiment, wireless I/O control device 226 is a media system remote control unit, which communicates with the components of the entertainment system 200 through IR signals. In yet another embodiment, wireless I/O control device 134 may be an IR remote control device similar in appearance to a typical entertainment system remote control with the added feature of a track-ball or other navigational mechanisms which allows a user to position a cursor on a display of the entertainment system 100.

[0030] The entertainment system 200 also includes a system controller 228. According to one exemplary embodiment of the present invention, the system controller 228 operates to store and display entertainment system data available from a plurality of entertainment system data sources and to control a wide variety of features associated with each of the system components. As shown in Figure 3, system controller 228 is coupled, either directly or indirectly, to each of the system components, as necessary, through I/O bus 210. In one exemplary embodiment, in addition to or in place of I/O bus 210, system controller 228 is configured with a wireless communication transmitter (or transceiver), which is capable of communicating with the system components via IR signals or RF signals. Regardless of the control medium, the system controller 228 is configured to control the media components of the media system 200 via a graphical user interface described below.

[0031] As further illustrated in Figure 3, media system 200 may be configured to receive media items from various media sources and service providers. In this exemplary embodiment, media system 200 receives media input from and, optionally, sends information to, any or all of the following sources: cable broadcast 230, satellite broadcast 232 (e.g., via a

satellite dish), very high frequency (VHF) or ultra high frequency (UHF) radio frequency communication of the broadcast television networks 234 (e.g., via an aerial antenna), telephone network 236 and cable modem 238 (or another source of Internet content). Those skilled in the art will appreciate that the media components and media sources illustrated and described with respect to Figure 3 are purely exemplary and that media system 200 may include more or fewer of both.

[0032] Figure 4 is a block diagram illustrating an embodiment of an exemplary system controller 228 according to the present invention. System controller 228 can, for example, be implemented as a set-top box and includes, for example, a processor 300, memory 302, a display controller 304, other device controllers (e.g., associated with the other components of system 200), one or more data storage devices 308 and an I/O interface 310. These components communicate with the processor 300 via bus 312. Those skilled in the art will appreciate that processor 300 can be implemented using one or more processing units. Memory device(s) 302 may include, for example, DRAM or SRAM, ROM, some of which may be designated as cache memory, which store software to be run by processor 300 and/or data usable by such programs, including software and/or data associated with the graphical user interfaces described below. Display controller 304 is operable by processor 300 to control the display of monitor 212 to, among other things, display GUI screens and objects as described below. Zoomable GUIs according to exemplary embodiments of the present invention provide resolution independent zooming, so that monitor 212 can provide displays at any resolution. Device controllers 306 provide an interface between the other components of the media system 200 and the processor 300. Data storage 308 may include one or more of

a hard disk drive, a floppy disk drive, a CD-ROM device, or other mass storage device.

Input/output interface 310 may include one or more of a plurality of interfaces including, for example, a keyboard interface, an RF interface, an IR interface and a microphone/speech interface. According to one exemplary embodiment of the present invention, I/O interface 310 will include an interface for receiving location information associated with movement of a wireless pointing device.

[0033] Generation and control of a graphical user interface according to exemplary embodiments of the present invention to display media item selection information is performed by the system controller 228 in response to the processor 300 executing sequences of instructions contained in the memory 302. Such instructions may be read into the memory 302 from other computer-readable mediums such as data storage device(s) 308 or from a computer connected externally to the media system 200. Execution of the sequences of instructions contained in the memory 302 causes the processor to generate graphical user interface objects and controls, among other things, on monitor 212. In alternative embodiments, hard-wire circuitry may be used in place of or in combination with software instructions to implement the present invention. Thus, the present invention is not limited to any specific combination of hardware circuitry and software.

[0034] Having described an exemplary media system which can be used to implement zoomable graphical interfaces according to the present invention, several examples of such interfaces will now be described. Referring first to Figures 5-8, an exemplary zoomable graphical user interface according to an exemplary embodiment of the present invention is described for use in displaying and selecting musical media items. Figure 5 portrays the

zoomable GUI at its most zoomed out state. Therein, the interface displays a set of shapes 500. Displayed within each shape 500 are text 502 and/or a picture 504 that describe the group of media item selections accessible via that portion of the GUI. As shown in Figure 5, the shapes 500 are rectangles, and text 502 and/or picture 504 describe the genre of the media. However, those skilled in the art will appreciate that this first viewed GUI grouping could represent other aspects of the media selections available to the user e.g., artist, year produced, area of residence for the artist, length of the item, or any other characteristic of the selection. Also, the shapes used to outline the various groupings in the GUI need not be rectangles. Shrunk down versions of album covers and other icons could be used to provide further navigational hints to the user in lieu of or in addition to text 502 and/or picture 504 within the shape groupings 500. A background portion of the GUI 506 can be displayed as a solid color or be a part of a picture such as a map to aid the user in remembering the spatial location of genres so as to make future uses of the interface require less reading. The selection pointer (cursor) 508 follows the movements of an input device and indicates the location to zoom in on when the user presses the button on the device (not shown in Figure 5).

[0035] According to one exemplary embodiment of applying the present invention to a PC, the input device can be a wireless mouse, e.g., the wireless mouse manufactured by Gyration, Inc. 12930 Saratoga Avenue, Bldg.C, Saratoga, CA 95070, coupled with a graphical user interface that support the point, click, scroll, hover and zoom building blocks. One feature of this particular input device that is beneficial for use in conjunction with the present invention is that it has only two buttons and a scroll wheel, i.e., three input actuation objects. One of the buttons can be configured as a ZOOM IN button and one can be configured as a

ZOOM OUT button. Compared with the conventional remote control units, e.g., that shown in Figure 1, the present invention simplifies this aspect of the GUI by greatly reducing the number of buttons, etc., that a user is confronted with in making his or her media item selection. Those skilled in the art will appreciate, however that any suitable input device can be used in conjunction with zoomable GUIs according to the present invention.

[0036] Figure 6 shows a zoomed in view of Genre 3 that would be displayed if the user selects Genre 3 from Figure 1, e.g., by moving the cursor 508 over the area encompassed by the rectangle surrounding Genre 3 on display 212 and depressing a button on the input device. The interface can animate the zoom from Figure 5 to Figure 6 so that it is clear to the user that a zoom occurred. An example of such an animated zoom/transition effect is described below. Once the shape 516 that contains Genre 3 occupies most of the screen on display 212, the interface reveals the artists 518 that have albums in the genre. In this example, seven different artists and/or their works are displayed. The unselected genres 515 that were adjacent to Genre 3 in the zoomed out view of Figure 5 are still adjacent to Genre 3 in the zoomed in view, but are clipped by the edge of the display 212. These unselected genres can be quickly navigated to by selection of them with selection pointer 508. Each of the artist groups, e.g., group 512, can contain images of shrunk album covers, a picture of the artist or customizable artwork by the user in the case that the category contains playlists created by the user.

[0037] A user may then select one of the artist groups for further review and/or selection. Figure 7 shows a further zoomed in view in response to a user selection of Artist 3 via positioning of cursor 508 and actuation of the input device, in which images of album

covers 520 come into view. As with the transition from the GUI screen of Figure 5 and Figure 6, the unselected, adjacent artists (artists #2, 6 and 7 in this example) are shown towards the side of the zoomed in display, and the user can click on these with selection pointer 508 to pan to these artist views. In this portion of the interface, in addition to the images 520 of album covers, artist information 524 can be displayed as an item in the artist group. This information may contain, for example, the artist's picture, biography, trivia, discography, influences, links to web sites and other pertinent data. Each of the album images 520 can contain a picture of the album cover and, optionally, textual data. In the case that the album image 520 includes a user created playlist, the graphical user interface can display a picture which is selected automatically by the interface or preselected by the user.

[0038] Finally, when the user selects an album cover image 520 from within the group 521, the interface zooms into the album cover as shown in Figure 8. As the zoom progresses, the album cover can fade or morph into a view that contains items such as the artist and title of the album 530, a list of tracks 532, further information about the album 536, a smaller version of the album cover 528, and controls 534 to play back the content, modify the categorization, link to the artists web page, or find any other information about the selection. Neighboring albums 538 are shown that can be selected using selection pointer 508 to cause the interface to bring them into view. This final zoom provides an example of semantic zooming, wherein certain GUI elements are revealed that were not previously visible at the previous zoom level. Various techniques for performing semantic zooming according to exemplary embodiments of the present invention are provided below.

[0039] As illustrated in the Figures 5-8 and the description, this exemplary embodiment of a graphical user interface provides for navigation of a music collection. Interfaces according to the present invention can also be used for video collections such as for DVDs, VHS tapes, other recorded media, video-on-demand, video segments and home movies. Other audio uses include navigation of radio shows, instructional tapes, historical archives, and sound clip collections. Print or text media such as news stories and electronic books can also be organized and accessed using this invention.

[0040] As will be apparent to those skilled in the art from the foregoing description, zoomable graphical user interfaces according to the present invention provide users with the capability to browse a large (or small) number of media items rapidly and easily. This capability is attributable to many characteristics of Applicants' interfaces including, but not limited to: (1) the use of images as all or part of the selection information for a particular media item, (2) the use of zooming to rapidly provide as much or as little information as a user needs to make a selection and (3) the use of several GUI techniques which combine to give the user the sense that the entire interface resides on a single plane, such that navigation of the GUI can be accomplished, and remembered, by way of the user's sense of direction. This latter aspect of GUIs according to the present invention is accomplished by, among other things, linking the various GUI screens together "geographically" by maintaining as much GUI object continuity from one GUI screen to the next, e.g., by displaying edges of neighboring, unselected objects around the border of the current GUI screen. As used in this text, the phrase "GUI screen" refers to a set of GUI objects rendered on one or more display units at the same time. A GUI screen may be rendered on the same display which outputs

media items, or it may be rendered on a different display. The display can be a TV display, computer monitor or any other suitable GUI output device.

[0041] Another GUI effect which enhances the user's sense of GUI screen connectivity is the panning animation effect which is invoked when a zoom is performed or when the user selects an adjacent object at the same zoom level as the currently selected object. Returning to the example of Figure 5, as the user is initially viewing this GUI screen, his or her point-of-view is centered about point 550. However, when he or she selects Genre 3 for zooming in, his or her point-of-view will shift to point 552. According to exemplary embodiments of the present invention, the zoom in process is animated to convey the shifting the POV center from point 550 to 552. This panning animation can be provided for every GUI change, e.g., from a change in zoom level or a change from one object to another object on the same GUI zoom level. Thus if, for example, a user situated in the GUI screen of Figure 6 selected the leftmost unselected genre 515 (Genre 2), a panning animation would occur which would give the user the visual impression of "moving" left or west. Applicants have discovered that providing a consistent sense of directional movement between GUI screens enables users to more rapidly navigate the GUI, both between zoom levels and between media items at the same zoom level.

[0042] Various data structures and algorithms can be used to implement zoomable GUIs according to the present invention. For example, data structures and algorithms for panning and zooming in an image browser which displays photographs have been described, for example, in the article entitled "Quantum Treemaps and Bubblemaps for a Zoomable Image Browser" to Benjamin B. Bederson, *UIST 2001, ACM Symposium on User Interface*

Software and Technology, CHI Letters, 3(2), pp. 71-80, the disclosure of which is incorporated here by reference. However, in order to provide a GUI for media selection which can, at a high level, switch between numerous applications and, at a lower level, provide user controls associated with selected images to perform various media selection functions, Applicants have discovered that additional data structures and algorithms are needed.

[0043] Zoomable GUIs can be conceptualized as supporting panning and zooming around a scene of user interface components in the view port of a display device. To accomplish this effect, zoomable GUIs according to exemplary embodiments of the present invention can be implemented using scene graph data structures. Each node in the scene graph represents some part of a user interface component, such as a button or a text label or a group of interface components. Children of a node represent graphical elements (lines, text, images, etc.) internal to that node. For example, an application can be represented in a scene graph as a node with children for the various graphical elements in its interface. Two special types of nodes are referred to herein as cameras and layers. Cameras are nodes that provide a view port into another part of the scene graph by looking at layer nodes. Under these layer nodes user interface elements can be found. Control logic for a zoomable interface programmatically adjusts a cameras view transform to provide the effect of panning and zooming.

[0044] Figure 9 shows a scene graph that contains basic zoomable interface elements which can be used to implement exemplary embodiments of the present invention, specifically it contains one camera node 900 and one layer node 902. The dotted line between

the camera node 900 and layer node 902 indicates that the camera node 900 has been configured to render the children of the layer node 902 in the camera's view port. The attached display device 904 lets the user see the camera's view port. The layer node 902 has three children nodes 904 that draw a circle and a pair of ovals. The scene graph further specifies that a rectangle is drawn within the circle and three triangles within the rectangle by way of nodes 912-918. The scene graph is tied into other scene graphs in the data structure by root node 920. Each node 906-918 has the capability of scaling and positioning itself relative to its parent by using a local coordinate transformation matrix. Figures 10(a) and 10(b) illustrate what the scene graph appears like when rendered through the camera at a first, zoomed out level of magnification and a second, zoomed in level of magnification, respectively.

[0045] Rendering the scene graph can be accomplished as follows. Whenever the display 914 needs to be updated, e.g., when the user triggers a zoom-in from the view of Figure 10(a) to the view of Figure 10(b), a repaint event calls the camera node 900 attached to the display 914 to render itself. This, in turn, causes the camera node 900 to notify the layer node 902 to render the area within the camera's view port. The layer node 900 renders itself by notifying its children to render themselves, and so on. The current transformation matrix and a bounding rectangle for the region to update is passed at each step and optionally modified to inform each node of the proper scale and offset that they should use for rendering. Since the scene graphs of applications operating within zoomable GUIs according to the present invention may contain thousands of nodes, each node can check the transformation matrix and the area to be updated to ensure that their drawing operations will

indeed be seen by the user. Although the foregoing example, describes a scene graph including one camera node and one layer node, it will be appreciated that exemplary embodiments of the present invention can embed multiple cameras and layers. These embedded cameras can provide user interface elements such as small zoomed out maps that indicate the user's current view location in the whole zoomable interface, and also allow user interface components to be independently zoomable and pannable.

[0046] When using a zoomable interface to coordinate the operation of multiple applications, e.g., like the movie and music browser described below with respect to Figures 14-18, the memory and resource requirements for each application may exceed the total memory available in the media system. This suggests that applications unload some or all of their code and data when the user is no longer viewing them. However, in zoomable GUIs according to the present invention it can be desirable to provide the appearance that some or all of the applications appear active to the user at all times. To satisfy these two competing objectives, the applications which are "off-screen" from the user's point of view can be put into a temporarily suspended state such that the user doesn't notice. To achieve this behavior in zoomable GUIs according to exemplary embodiments of the present invention, events are sent to applications to indicate when they enter and exit a view port. One way to implement such events is to add logic to the code that renders a component so that it detects when the user enters a view port. However, this implies that the notification logic gets invoked at every rendering event and, more importantly, that it cannot easily detect when the user has navigated the view port away from the component. Another method for sending events to applications is to incorporate the notification logic into the GUI navigation elements (such as

hyperlinks and buttons), so that they send notifications to the component when they change the view port of a camera to include the component of interest. However, this requires the programmer to vigilantly add notification code to all possible navigation UI elements.

[0047] According to one exemplary embodiment, a computationally efficient node watcher algorithm can be used to notify applications regarding when GUI components and/or applications enter and exit the view of a camera. At a high level, the node watcher algorithm has three main processing stages: (1) initialization, (2) view port change assessment and (3) scene graph change assessment. The initialization stage computes node quantities used by the view port change assessment stage and initializes appropriate data structures. The view port change assessment stage gets invoked when the view port changes and notifies all watched nodes that entered or exited the view port. Finally, the scene graph change assessment stage updates computations made at the initialization stage that have become invalid due to changes in the scene graph. For example, if an ancestor node of the watched node changes location in the scene graph, computations made at initialization may need to be recomputed.

[0048] Of these stages, view port change assessment drives the rest of the node watcher algorithm. To delineate when a node enters and exits a view port, the initialization step determines the bounding rectangle of the desired node and transforms it from its local coordinate system to the local coordinate system of the view port. In this way, checking node entrance does not require a sequence of coordinate transformations at each view port change. Since the parents of the node may have transform matrices, this initialization step requires traversing the scene graph from the node up to the camera. If embedded cameras are used in

the scene graph data structure, then multiple bounding rectangles may be needed to accommodate the node appearing in multiple places.

[0049] Once the bounding rectangle for each watched node has been computed in the view port coordinate system, the initialization stage adds the bounding rectangle to the view port change assessment data structures. The node watcher algorithm uses a basic building block for each dimension in the scene. In zoomable interfaces according to some exemplary embodiments, this includes an x dimension, a y dimension, and a scale dimension. As described below, however, other exemplary implementations may have additional or different dimensions. The scale dimension describes the magnification level of the node in the view port and is described by the following equation:

$$s = \frac{d'}{d}$$

Where s is the scale, d is the distance from one point of the node to another in the node's local coordinates and d' is the distance from that point to the other in the view port.

[0050] Figure 11 shows an exemplary building block for detecting scene entrance and exit in one dimension. The following describes handling in the x dimension, but those skilled in the art will appreciate that the other dimensions can be handled in a similar manner. The Region Block 1100 contains references to the transformed bounding rectangle coordinates. This includes the left and right (top and bottom or minimum and maximum scale) offsets of the rectangle. The left and right offsets are stored in Transition Blocks 1102 and 1104, respectively, that are themselves placed in an ordered doubly linked list, such that lower

numbered offsets are towards the beginning. The current view port bounds are stored in the View Bounds block 1106. Block 1106 has pointers to the Transition Blocks just beyond the left and right side of the view, e.g., the Transition Block immediately to the right of the one pointed to by View Left Side is in the view unless that latter block is pointed to by View Right Side.

[0051] When the view port changes, the following processing occurs for each dimension. First, the View Left Side and View Right Side pointers are checked to see if they need to be moved to include or exclude a Transition Block. Next, if one or both of the pointers need to be moved, they are slid over the Transition Block list to their new locations. Then, for each Transition Block passed by the View Left Side and View Right Side pointers, the node watcher algorithm executes the Transition Block notification code described below. This notification code determines if it is possible that its respective node may have entered or exited the view port. If so, that node is added to a post processing list. Finally, at the end of this processing for each dimension, each node on the post processing list is checked that its view port status actually did change (as opposed to changing and then changing back). If a change did occur, then the algorithm sends an event to the component. Note that if the view port jumps quickly to a new area of the zoomable interface that the algorithm may detect more spurious entrance and exit events.

[0052] The Transition Block notification code can be implemented as a table lookup that determines whether the node moved into or out of the view port for the dimension being checked. An exemplary table is shown below.

Node side	View side	View Move Direction	Partial Intersection Notification	Full Intersection Notification
Left	Left	Left	None	Enter
Left	Left	Right	None	Exit
Right	Left	Left	Enter	None
Right	Left	Right	Exit	None
Left	Right	Left	Exit	None
Left	Right	Right	Enter	None
Right	Right	Left	None	Exit
Right	Right	Right	None	Enter

Table 1 - Transition Notification Table

Columns 1, 2 and 3 are the inputs to the Transition Notification Table. Specifically, the node watcher algorithm addresses the table using a combination of the node side, view side and view move direction to determine whether the node being evaluated was entered, exited or not impacted. Column 1 refers to the side of the node represented by the Transition Block that was passed by the view port pointers. Column 2 refers to the side of the view port and column 3 refers to the direction that that side of the view port was moving when it passed the node's Transition Block. Either output column 4 or 5 is selected depending upon whether the node should be notified when it is partially or fully in view. For example, in some implementations it may be desirable to notify an application such as a streaming video window only after it is fully in view since loading a partially-in-view video window into the zoomable GUI may be visually disruptive.

[0053] When the output of the table indicates enter or exit, the node watcher algorithm adds the node to the post processing list. The output columns of Table 1 are populated based on the following rules. If the node intersects in all dimensions then an enter notification will be sent in the post processing step. If the node was in the view and now one or more dimensions have stopped intersecting, then an exit notification will be sent. To

reduce the number of nodes in the post processing list, the Transition Block notification code checks for intersection with other dimensions before adding the node to the list. This eliminates the post processing step when only one or two dimensions out of the three intersect.

[0054] In addition to using node watcher notifications for application memory management, this algorithm can also be used for other functions in zoomable GUIs according to the present invention. For example, the node watcher algorithm can be used to change application behavior based on the user's view focus, e.g., by switching the audio output focus to the currently viewed application. Another application for the node watcher algorithm is to load and unload higher resolution and composite images when the magnification level changes. This reduces the computational load on the graphics renderer by having it render fewer objects whose resolution more closely matches the display. In addition to having the node watcher algorithm watch a camera's view port, it is also useful to have it watch the navigation code that tells the view port where it will end up after an animation. This provides earlier notification of components that are going to come into view and also enables zoomable GUIs according to exemplary embodiments of the present invention to avoid sending notifications to nodes that are flown over due to panning animations.

[0055] In addition to the node watcher algorithm described above, Applicants have developed resolution independent semantic zooming algorithms which can be used in zoomable GUIs according to exemplary embodiments of the present invention. Semantic zooming refers to adding, removing or changing details of a component in a zoomable GUI depending on the magnification level of that component. For example, in the movie browser

interface described below, when the user zooms close enough to the image of the movie, it changes to show movie metadata and playback controls. The calculation of the magnification level is based on the number of pixels that the component uses on the display device. The zoomable GUI can store a threshold magnification level which indicates when the switch should occur, e.g., from a view without the movie metadata and playback controls to a view with the movie metadata and playback controls.

[0056] Television and computer displays have widely varying display resolutions. Some monitors have such a high resolution that graphics and text that is readable on a low resolution display is so small to become completely unreadable. This also creates a problem for applications that use semantic zooming, especially on high resolution displays such as HDTVs. In this environment, semantic zooming code that renders based on the number of pixels displayed will change the image before the more detailed view is readable. Programmatically modifying the threshold at which semantic zooming changes component views can only work for one resolution.

[0057] The desirable result is that semantic zooming occurs consistently across all monitor resolutions. One solution is to use lower resolution display modes on high resolution monitors, so that the resolution is identical on all displays. However, the user of a high resolution monitor would prefer that graphics would be rendered at their best resolution if semantic zooming would still work as expected. Accordingly, Applicants have developed a semantic zooming technique which supports displays of all different resolutions without the previously stated semantic viewing issues. This can be accomplished by, for example, creating a virtual display inside of the scene graph. This is shown in Figure 12 by using an

embedded virtual camera node 1200 and adding logic to compensate for the display resolution. The virtual camera node 1200 defines a view port whose size maps to the user's view distance and monitor size. For example, a large virtual camera view port indicates that a user is either sitting close enough to the monitor or has a large enough monitor to resolve many details. Alternately, a small view port indicates that the user is farther away from the monitor and requires larger fonts and image. The zoomable GUI code can base the semantic zooming transitions on the magnification level of components seen on this virtual camera and using the user's preferred viewing conditions.

[0058] The main camera node 1202 that is attached to the display device 1204 has its view port configured so that it displays everything that the virtual camera 1200 is showing. Since graphics images and text are not mapped to pixels until this main camera 1202, no loss of quality occurs from the virtual camera. The result of this is that high definition monitors display higher quality images and do not trigger semantic zooming changes that would make the display harder to read.

[0059] According to one exemplary embodiment of the present invention, the process works as follows. Each camera and node in the scene graph has an associated transform matrix (T_1 to T_n). As shown in Table 1, these matrices transform that node's local coordinate system to that of the next node towards the display. In the figure, T_1 transforms coordinates from its view port to display coordinates. Likewise, T_2 transforms its local coordinate system to the camera's view port. If the leaf node 1206 needs to render something on the display, it computes the following transform matrix:

$$A = T_1 T_2 \cdots T_n$$

This calculation can be performed while traversing the scene graph. Since the component changes to support semantic zooming are based on the virtual camera 1200, the following calculation is performed:

$$B = T_4 T_5 \cdots T_n$$

Typically, T_1 to T_3 can be determined ahead of time by querying the resolution of the monitor and inspecting the scene graph. Determining B from A is, therefore, accomplished by inverting these matrices and multiplying as follows:

$$B = (T_1 T_2 T_3)^{-1} A$$

For the case when calculating T_1 to T_3 ahead of time is problematic, e.g., if a graphics API hides additional transformations, logic can be added to the virtual camera to intercept the transformation matrix that it would have used to render to the display. This intercepted transformation is then inverted and multiplied as above to compute the semantic zooming threshold.

[0060] One strength of zoomable interfaces according to exemplary embodiments of the present invention is the ability to maintain context while navigating the interface. All of the interface components appear to exist in the zoomable world, and the user just needs to pan and zoom to reach any of them. The semantic zooming technique described above changes the appearance of a component depending on the zoom or magnification level. Figures 13(a) and 13(b) provide an example of semantic zooming for a component where the zoomed out version of the component (Figure 13(a)) is a picture and the zoomed in version (Figure 13(b))

includes the same picture as well as some controls and details. Some more detailed examples of this are provided below. One challenge associated with semantic zooming is that changes between views can occur abruptly, and transition techniques such as alpha blending do not provide visually pleasing results when transitioning between two such views.

[0061] Accordingly, exemplary embodiments of the present invention provide for some common image or text in all views of a component to provide a focal point for a transition effect when a semantic zoom is performed. For example, in Figures 13(a) and 13(b), the common element is the picture. The transition effect between the zoomed out version and the zoomed in version can be triggered using, for example, the above-described node watcher algorithm as follows. First, a registration with the node watcher can be performed to receive an event when the main camera's view port transitions from the magnification level of the zoomed out version of the component to the zoomed in version. Then, when the event occurs, an animation can be displayed which shows the common element(s) shrinking and translating from their location in the zoomed out version to their location in the zoomed in version. Meanwhile, the camera's view port continues to zoom into the component.

[0062] These capabilities of graphical user interfaces according to the present invention will become even more apparent upon review of another exemplary embodiment described below with respect to Figures 14-18. Therein, a startup GUI screen 1400 displays a plurality of icons which operate as media group representations. The purely exemplary media group representations of home video, movies, TV, sports, radio, music and news could, of course include different, more or fewer media group representations. Upon actuation of one

of these icons by a user, the GUI according to this exemplary embodiment will then display a plurality of images each grouped into a particular category or genre. For example, if the “movie” icon in Figure 14 was actuated by a user, the GUI screen of Figure 15 can then be displayed. Therein, it can be seen that a 120 images are displayed, each of which is categorized in a particular group, specifically action, classics, comedy, drama, family and new releases. Those skilled in the art will appreciate that more or fewer categories could be provided. In this exemplary embodiment, the media item images are cover art associated with each movie selection. For this exemplary embodiment, the level of magnification of the images is such that the identity of the movie can readily be discerned by its associated image, although much of the text may be too small to easily read.

[0063] The cursor (not shown in Figure 15) can then be disposed over a group of the movie images and the input device actuated to provide a selection indication for one of the groups. In this example the user selects the drama group and the graphical user interface then displays a zoomed version of the drama group of images as seen in Figure 16. As with the previous embodiment, a transition effect can also be displayed as the GUI shifts from the GUI screen of Figure 15 to the GUI screen of Figure 14, e.g., the GUI may pan the view from the center of the GUI screen of Figure 15 to the center of the drama group of images during or prior to the zoom. Note that although the zoomed version of the drama group of Figure 16 only displays a subset of the total number of images in the drama group, that this zoomed version can alternatively contain all of the images in the selected group. The choice of whether or not to display all of the images in a selected group in any given zoomed in version of a GUI screen can be made based upon, for example, the number of media items in a group

and a minimum desirable magnification level for a media item for a particular zoom level. This latter characteristic of GUIs according to the present invention can be predetermined by the system designer/service provider or can be user customizable via software settings in the GUI. For example, the number of media items in a group and the minimum and/or maximum magnification levels can be configurable by either or both of the service provider or the end user. Such features enable those users with, for example, poor eyesight, to increase the magnification level of media items being displayed. Conversely, users with especially keen eyesight may decrease the level of magnification, thereby increasing the number of media items displayed on a GUI screen at any one time and decrease browsing time.

[0064] One exemplary transition effect which can be employed in graphical user interfaces according to the present invention is referred to herein as the “shoe-to-detail” view effect. When actuated, this transition effect takes a zoomed out image and simultaneously shrinks and translates the zoomed out image into a smaller view, i.e., the next higher level of magnification. The transition from the magnification level used in the GUI screen of Figure 15 to the greater magnification level used in the GUI screen of Figure 16 results in additional details being revealed by the GUI for the images which are displayed in the zoomed in version of Figure 16. The GUI selectively reveals or hides details at each zoom level based upon whether or not those details would display well at the currently selected zoom level. Unlike a camera zoom, which attempts to resolve details regardless of their visibility to the unaided eye, exemplary embodiments of the present invention provide for a configurable zoom level parameter that specifies a transition point between when to show the full image and when to show a version of the image with details that are withheld. The transition point

can be based upon an internal resolution independent depiction of the image rather the resolution of TV/Monitor 212. In this way, GUIs according to the present invention are consistent regardless of the resolution of the display device being used in the media system.

[0065] In this exemplary embodiment, an additional amount of magnification for a particular image can be provided by passing the cursor over a particular image. This feature can be seen in Figure 17, wherein the cursor has rolled over the image for the movie "Apollo 13". Therein it can be seen that this image has been magnified to make more legible the quote "Houston, we have a problem" relative to the corresponding image in the GUI screen of Figure 16. User selection of this image, e.g., by depressing a button on the input device, can result in a further zoom to display the details shown in Figure 18. This provides yet another example of semantic zooming as it was previously described since various information and control elements are revealed in the GUI screen of Figure 18 that were not visible in the GUI screen of Figure 17. For example, information about the movie "Apollo 13" including, among other things, the movie's runtime, price and actor information is shown. Those skilled in the art will appreciate that other types of information could be provided here. Additionally, this GUI screen includes GUI control objects including, for example, button control objects for buying the movie, watching a trailer or returning to the previous GUI screen (which could also be accomplished by depressing the ZOOM OUT button on the input device). Hyperlinks can also be used to allow the user to jump to, for example, GUI screens associated with the related movies identified in the lower right hand corner of the GUI screen of Figure 18 or information associated with the actors in this movie. In this example, some or all of the film titles under the heading "Filmography" can be implemented as hyperlinks which, when

actuated by the user via the input device, will cause the GUI to display a GUI screen corresponding to that of Figure 18 for the indicated movie.

[0066] A transition effect can also be employed when a user actuates a hyperlink.

Since the hyperlinks may be generated at very high magnification levels, simply jumping to the linked media item may cause the user to lose track of where he or she is in the media item selection "map". Accordingly, exemplary embodiments of the present invention provide a transition effect to aid in maintaining the user's sense of geographic position when a hyperlink is actuated. One exemplary transition effect which can be employed for this purpose is a hop transition. In an initial phase of the transition effect, the GUI zooms out and pans in the direction of the item pointed to by the hyperlink. Zooming out and panning continues until both the destination image and the origination image are viewable by the user.

Using the example of Figure 18 once again, if the user selects the hyperlink for "Saving Private Ryan", then the first phase of the hyperlink hop effect would include zooming out and panning toward the image of "Saving Private Ryan" until both the image for "Saving Private Ryan" and "Apollo 13" were visible to the user. At this point, the transition effect has provided the user with the visual impression of being moved upwardly in an arc toward the destination image. Once the destination image is in view, the second phase of the transition effect gives the user the visual impression of zooming in and panning to, e.g., on the other half of the arc, the destination image. The hop time, i.e., the amount of time both phases one and two of this transition effect are displayed, can be fixed as between any two hyperlinked image items. Alternatively, the hop time may vary, e.g., based on the distance traveled over the GUI. For example, the hop time can be parameterized as $\text{HopTime} = A \log(\text{zoomed-in})$

scale level/hop apex scale level) + B(distance between hyperlinked media items) + C, where A, B and C are suitably selected constant values.

[0067] The node watcher algorithm described above with respect to Figures 9-11 can also be used to aid in the transition between the zoom level depicted in the exemplary GUI screen of Figure 17 and the exemplary GUI screen of Figure 18. The rendering of GUI screens containing text and/or control elements which are not visible in other zoom level versions of the selected image may be more computationally and/or memory intensive than the images at lower magnification levels. Accordingly, the node watcher algorithm can be used in exemplary embodiments of the present invention to aid in pre-loading of GUI screens such as that shown in Figure 18 by watching the navigation code of the GUI to more rapidly identify the particular media item being zoomed in on.

[0067] Having provided some examples of zoomable graphical user interfaces according to the present invention, exemplary frameworks and infrastructures for using such interfaces will now be described. Figure 19 provides a framework diagram wherein zoomable interfaces associated with various high level applications 1900, e.g., movies, television, music, radio and sports, are supported by primitives 1902 (referred to in the Figure as "atoms"). In this exemplary embodiment, primitives 1902 include POINT, CLICK, ZOOM, HOVER and SCROLL, although those skilled in the art will appreciate that other primitives may be included in this group as well, e.g., PAN. As described above the POINT and CLICK primitives operate to determine cursor location and trigger an event when, for example, a user actuates the ZOOM IN or ZOOM OUT button on the handheld input device. These primitives simplify navigation and remove the need for repeated up-down-right-left button

actions. As illustrated above, the ZOOM primitive provides an overview of possible selections and gives the user context when narrowing his or her choices. This concept enables the interface to scale to large numbers of media selections and arbitrary display sizes. The SCROLL primitive handles input from the scroll wheel input device on the exemplary handheld input device and can be used to, for example, accelerates linear menu navigation. The HOVER primitive dynamically enlarges the selections underneath the pointer to enable the user to browse potential choices without committing.

[0068] Between the lower level primitives 1902 and the upper level applications 1900 reside various software and hardware infrastructures 1904 which are involved in generating the images associated with zoomable GUIs according to the present invention. As seen in Figure 19, such infrastructures 1904 can include a handheld input device/pointer, application program interfaces (APIs), zoomable GUI screens, developers' tools, etc. Another set of infrastructures according to exemplary embodiments of the present invention, is seen in Figure 20, wherein a metaserver 2000, a local system controller 2002 (e.g., a set-top box) and a handheld input device 2004 communicate with one another to generate zoomable GUIs 2006 according to exemplary embodiments of the present invention. The metadata server 2000 can store all of the images, metadata, and purchasing information needed for displaying the movies, music and other content in the zoomable GUI 2006. The metadata server 2000 also schedules metadata delivery over the headend network 2008 to minimize the downstream bandwidth used and processes requests from the local system controller 2002 for obscure metadata and purchases. The local system controller 2002 assimilates the metadata for presentation on the display device, manages user navigation

through the zoomable movie and music selection scenes, and coordinates content playback.

The local system controller 2002 may also store the user interface (UI) software which operates as described above to generate zoomable GUIs according to exemplary embodiments of the present invention. The handheld input device 2004 sends the pointing, clicking, and button events from the user to the local system controller 2006.

[0069] The foregoing exemplary embodiments are purely illustrative in nature. The number of zoom levels, as well as the particular information and controls provided to the user at each level may be varied. For example, instead of (or in addition to) the iconic application representations of Figure 14, a more detailed graphical filter mechanism can be provided. The filter mechanism permits the user to select search parameters which are used to generate a "results" GUI screen containing those media items (images) which match the search criteria. This provides another mechanism for scaling the large number of media selection items available to the media system for browsing. As an example, the filter mechanism may be displayed as a hierarchical menu of icons that filter movie selections by, e.g., title, actor, genre, year produced, rating, etc. When the user has finished specifying all of the desired filters, they can actuate a GUI control to generate and display the results, e.g., resulting in a display of all of the media selection items (images) associated with the drama genre. Any desired filter mechanism can be used, even a text entry approach which permits the user to enter information used to filter the collection of images which are used as media selection objects.

[0070] Those skilled in the art will appreciate that the present invention provides revolutionary techniques for presenting large and small sets of media items using a zoomable

interface such that a user can easily search through, browse, organize and play back media items such as movies and music. Graphical user interfaces according to the present invention organize media item selections on a virtual surface such that similar selections are grouped together. Initially, the interface presents a zoomed out view of the surface, and in most cases, the actual selections will not be visible at this level, but rather only their group names. As the user zooms progressively inward, more details are revealed concerning the media item groups or selections. At each zoom level, different controls are available so that the user can play groups of selections, individual selections, or go to another part of the virtual surface to browse other related media items. Zooming graphical user interfaces according to exemplary embodiments of the present invention can contain categories of images nested to an arbitrary depth as well as categories of categories.

[0071] The above-described exemplary embodiments are intended to be illustrative in all respects, rather than restrictive, of the present invention. Thus the present invention is capable of many variations in detailed implementation that can be derived from the description contained herein by a person skilled in the art. All such variations and modifications are considered to be within the scope and spirit of the present invention as defined by the following claims. No element, act, or instruction used in the description of the present application should be construed as critical or essential to the invention unless explicitly described as such. Also, as used herein, the article "a" is intended to include one or more items.